

On Network Flow Management in Virtualized Environments

Fernando Rodríguez-Haro, Felix Freitag, Leandro Navarro
*Computer Architecture Department
Polytechnic University of Catalonia
Barcelona Spain
Email: frodrigu@ac.upc.edu*

Nicandro Farías-Mendoza, Juan Antonio Guerrero-Ibáñez
*Faculty of FIME and Telematic
University of Colima
Colima México
Email: nfarias@ucol.mx*

Abstract—Virtualization has enabled the use of Virtual Machine (VM) based resource providers to offer computing power as a service. However, one important feature is to have certainty about the planned or expected performance of the applications that run inside the VMs. Therefore, consolidating the work load of web-based applications, with time-varying resource requirements, should be done in an automated way. We present and evaluate a network flow management component for the admission control of HTTP connections. We study this component, in our Local Resource Manager (LRM), by using two types of common web-based applications. The preliminary results shows that we can meet the number of successful served requests and at the same time avoid the degradation in the application's response times.

Keywords—management; virtualization

I. INTRODUCTION

Recently, web service applications are the common infrastructure to provide Software as a Service (SaaS) solutions. Examples of such applications range from general solutions such as Google's Business Solutions to specialized solutions such as scientific remote instrumentation. Despite the fact that web service applications are widespread used, their properties (response times, web based, erratic burst behavior) present a challenge for their autonomous resource management in virtualized environments.

The advent of virtualization solutions and the demanding of online computations requires specialized QoS-aware techniques to manage the underlying resources of the virtualization layer. A solution for the time-varying constraints of dynamic applications is an open challenge. A key problem is that different applications have different behaviors and requirements that can not be expressed in CPU rate terms.

In this work we present an approach in which the performance of a web service application is not only based on meeting an application-based Service Level Objective (SLO) such as serving a given number HTTP requests, but also on keeping control on the number of connections allowed to reach a web-based server.

We adopt an hybrid approach to develop the proposal of a connection-aware Local Resource Manager (LRM). Our approach provides autonomous CPU resource management enhanced with a policy-based connections manager for man-

aging the network flow of web-based applications. All of this in the context of virtualized environments.

This paper is organized as follows: In section 2 we discuss the related work to our approach. In section 3 we explain the connection-aware virtualization manager. In section 4 we report experiments in which we study the implemented component. Finally, section 6 concludes and provides an outlook on future work.

II. RELATED WORK

Kephart et al [1] summarize the importance of self-management systems in the context of autonomic computing. These systems accept high-level objectives from administrators and apply self management policies.

Policy-based QoS control and learning have been proposed in non-VM contexts. Solutions based on QoS guarantees have been discussed using control theory [2], [3], online analytic performance models [4], regression-based analytic models [5], and statistical inference [6]. Applications of these approaches are dynamic provisioning [7] and energy conservation [8]. In order to meet the QoS guarantees the approaches change the applications parameters and/or the OS scheduler parameters. However, these approaches are unaware of the virtualization layer that shares the resources among multiple VMs. Our approach aims to provide a framework to meet the VM requirements for the dynamic workload of the hosted application.

A Grid computing infrastructure, which is a particular form of distributed computing, can be used for the deployment of utility computing. The work [9] presents an approach based on Globus services and the GridWay workload manager. The Grid computing overcomes some of the utility computing challenges to provide computing resources. However, the usage of the servers is not homogeneous and leads to have underutilized nodes. This problem is caused by two factors, firstly the applications have different on demand requirements and secondly the computing resources are managed in a coarse-grain fashion (i.e. a pool of physical servers).

One solution that has revived the interest of Utility Computing is virtualization. VMware and Xen are well known

server virtualization solutions. The server virtualization solutions of Xen and VMware have proved to manage the transparent access to subsets of physical resources while assuring a high degree of VM isolation.

Virtualization can be used as a platform for provisioning systems in Grid environments. When we refer to a provisioning system we mean the software which aims to provide the global resource manager mechanisms that enable the infrastructure connection between resource consumers (end users or software agents) and resource providers. Therefore, a virtualized Grid infrastructure needs mechanisms to offer and request service levels in the resource provided.

III. CONNECTION-AWARE VIRTUALIZATION MANAGER

Menascé and Casalicchio [10] stated that planning the capacity to guarantee quality of service in large scale scenarios is a challenge because global service-level agreements (SLAs) depend on local SLAs. Colling et al [11] proposed a framework for supporting QoS guarantees via both reservation and discovery of best-effort services. They also discussed that applications with specific QoS requirements can benefit from mechanisms for the support of both strict and loose guarantees.

Thus, a new challenge is presented. *How to manage and assure local SLAs with strict and loose guarantees in the context of VM-based resource providers.* In order to address this problem, we use the primitives offered by the virtualization layer to build our enhanced management component which has the following features.

- 1) An inter-domain communication mechanism within the VM-based resource provider.
- 2) An automated component that collects the CPU metrics of the VM.
- 3) A non-intrusive application-helper component that collects the high-level metrics of the hosted applications.
- 4) An adjustment function to properly slice the CPU resources according to the acquired knowledge and the target QoS.

The controller component for the management of VM-based resource providers behaves like a self-tuning component. We have set to two the number of samples needed by the controller. The sensor is called each 2.5 secs, this leads to a controller period of 5 secs.

We have proposed an hybrid adapting algorithm that slices CPU resources according to low-level and high-level requirements [12]. The controller focus on meet a QoS requirement expressed in terms of CPU rate. This controller is enhanced with a learning phase. The learning phase is embedded in the controller but invoked at a different period. Each learning phase lasts 6 times controller-period, i.e. 50secs. We find out that this values is enough for a learning phase allows to change and adapt the original QoS requirement based on the measured application's performance and the target service level objective.

A. Network flow management

This component runs in the privileged domain and includes an admission control component and the application-helper to carry on the measurement of the web-related metrics. However, instead of launching an application-helper per virtual machine, we developed an application-helper that runs in the Domain-0. This process keeps track of each http request since it arrives until it is served. The sampling period to notify the metrics in the XenStore database is set to 2.5 secs, It runs independently from the (CPU metrics) sensor and the controller. This approach allows us to have a real-time metering solution that measures metrics such as: requests/s, accurate mean response times, arrival rates, and queued requests. Additionally, our application helper constructs distribution vectors for the response times of the served http requests. This distribution vectors allow users to express more realistic web requirements such as "serve 90% of http requests below a given response time". Finally, the admission control component keeps track of each VM connection and manages the arrival rate of the HTTP connections. Therefore, the external input is controlled according to the user's requirement and the capacity of the application.

IV. EXPERIMENTS

We evaluate our connection-aware manager with two typical web-based services: (1) a web-based server-side application, and (2) a web-based image server. Our goal is not only to manage the erratic behavior of the web-based services, but also to assure a user-perceived quality of service in the responses times of the servers. Additionally, the elected applications present different resource consumption behaviors, the former is CPU intensive and the latter is network intensive.

1) Execution context: For the experiments we use two physical machines with the same characteristics. Quad CPU Q6700 2.66GHz, 8GB of memory, 750GB SATA Disk, and Gigabit NIC. Both interconnected through a gigabit switch.

One physical machine is deployed with the connection-aware component and the Local Resource Manager. The operating system is Fedora Core 8 and for virtualization we use Xen 3.1.0. We deploy paravirtualized VMs with Fedora Core 8 for the guest OS.

The second physical machine is used to simulate the web client requests. For this task we use Siege [13] an http regression testing and benchmarking utility to stress web servers with a configurable number of concurrent simulated users.

We deploy the web-based server-side application in two virtual machines named pc1 and pc2. For this web-based service we use Image_3D [14] a highly object oriented PHP5 package that allows the rendering of 3 dimensional objects using PHP. Figure 1 shows the three services we used during the experiments.

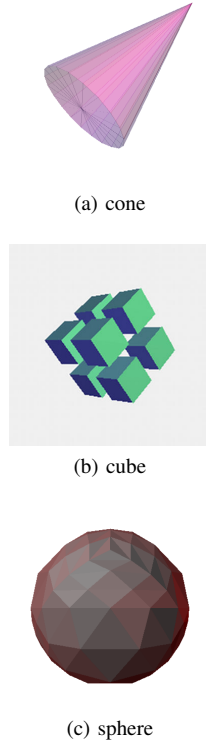


Figure 1. Rendered images by the application web server.

File	size (KB)
image1	177.4
image2	249.5
image3	250.9
image4	509.3

Table I
CHARACTERISTICS OF THE JPG IMAGES HOSTED IN THE VIRTUAL
MACHINE PC3

The web server on charge of serving jpg images is deployed in a third virtual machine named pc3. Whilst the dimension of the images is 1280x800, the file size is different as shown in Table I.

A. Base line experiment

In the first experiment we evaluate the web application server and in the second experiment we evaluate the web server in charge of serving jpg images. For both experiments we launch the benchmarking tool with an incremental load in the number of simulated web clients (2,8,32,64,128), each one with think-time=0 and a duration of 60secs.

The results shows the saturation points of both experiments for CPU and request served. The surges correspond to the start and end times of the load generated by the benchmarking tool, each one with a duration of 60secs. Figure 2 shows that the cost of rendering 3D objects impacts in the CPU consumption, in fact pc1 makes heavy use of the four processors reaching almost 95% of CPU consumption.

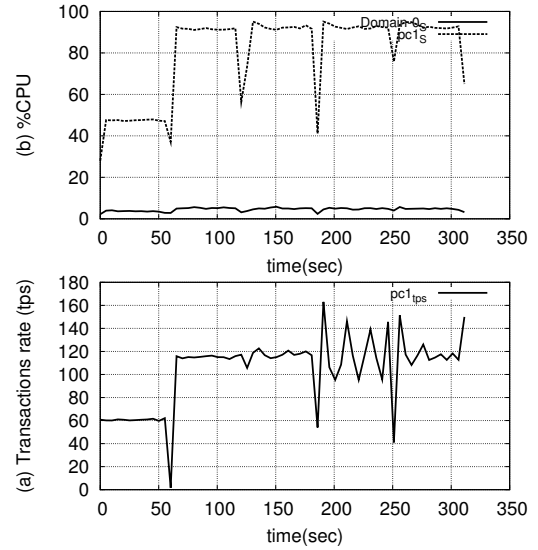


Figure 2. Application web server

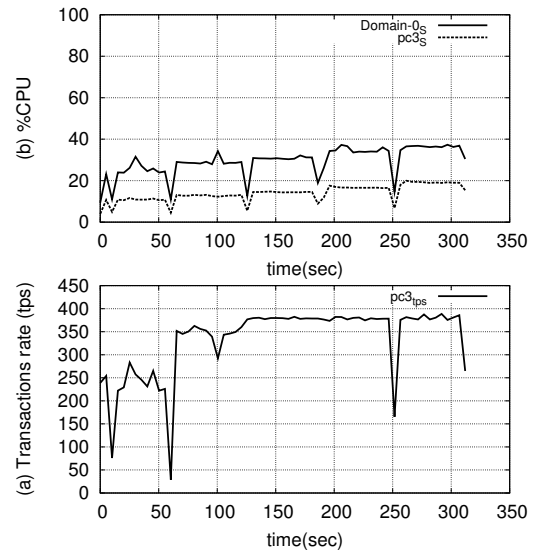


Figure 3. Web image server

On the other hand, Figure 3 shows that pc3 consumes much less CPU. In both cases, the requests achieved reach their saturation point at second 190.

In order to measure the accuracy of our instrumented measurement components we present Table II and Table III. The column RespTime is the mean response time perceived by the client, in our results we find out that the application-helper has a high precision in the reported metrics. Other important metric is the column Throughput, unlike the web application that consumes few network resources, serving jpg images makes heavy use of the network resource. Table III shows that 32 simulated users (with think-time = 0) can

Users	Trans	RespTime	TransRate	Throughput(MBps)	Failed	LongestTrans
2	3586	0.03	60.23	0.24	0	0.25
8	6945	0.07	115.79	0.37	0	0.41
32	6978	0.27	116.36	0.35	0	7.12
128	6897	1.11	115.01	0.36	0	4.47
256	6724	1.99	112.12	0.32	29	39.01

Table II
METRICS REPORTED BY THE BENCHMARKING TOOL FOR THE APPLICATION WEB SERVER

Users	Trans	RespTime	TransRate	Throughput (MBps)	Failed	LongestTrans
2	15074	0.01	251.91	73.01	0	3.01
8	21406	0.02	356.95	103.44	0	3.03
32	23031	0.08	383.98	111.26	0	3.31
128	22982	0.33	383.22	110.94	0	12.01
256	22945	0.66	382.54	110.62	0	29.06

Table III
METRICS REPORTED BY THE BENCHMARKING TOOL FOR THE WEB SERVER OF IMAGES

consume the 1Gb capacity of the network link.

B. Net Flow Management Evaluation

In this experiment we present our autonomous controller enhanced with a network QoS policy manager that prevents the saturation of the web applications. We achieve this by accepting only the number of connections that can be managed by the VM. This policy avoids that the VM can be reached and in consequence prevents its saturation. This means that the web server will meet not only the SLO requested but also with an expressed quality in the response times of the requests. For this experiment we benchmarked the web application in pc2 and the image server in pc3. Table IV shows the end-user inputs that vary through the length of this experiment.

Figure 4 shows the results of this experiment. In this experiment we see the controller changing the CPU resources of two web-based services which have different resource intensive requirements. Despite pc3 is network intensive with low CPU consumption and pc2 is CPU intensive with low network consumption, the controller is able to manage both type of applications. At the same time, it is achieved that the network QoS policy adjusts the acceptance rate in the number of incoming connections that can reach each VM. A closer look in Figure 4b shows that after time 100s more than 90% of the requests were served with a response time below one second.

V. CONCLUSIONS

In this work we evaluate our connection-aware local resource manager with two web service applications. The common behavior of these applications is the erratic burst behavior. Properties such as quality of service in the response times and transaction rate are the common metrics to

measure the responsiveness of these applications. Such characteristics present a challenge for developing autonomous resource management in virtualized environments.

We present a VM-based resource provider enabled with an automated component that collects the metrics of the VMs performance and the application's performance. With this information, a self-management algorithm meet the agreed requirements. Unlike hand-made configuration this self-managing approach allows expressing global objectives and eases the deployment of a VM-based resource provider in large scenarios. Additionally, the length of the sensor and controller period is fixed and they were obtained by comparing the converging times during the runs of several experiments. However, further research includes the ability to adjust these parameters according to the dynamic behavior of the target application.

The results show that we can better manage physical resources by meeting the requirements of all VMs while allowing the dynamic adjustment of the CPU assigned. Additionally we achieved to manage not only the requested SLO but also the controller accomplish a quality in the user-perceived response time.

For future work we plan to test and tune our approach in the context of database application. Specifically, we want to explore our local resource manager in the context of a dynamic database cluster.

REFERENCES

- [1] J. Kephart and D. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.
- [2] T. Abdelzaher, J. Stankovic, C. Lu, R. Zhang, Y. T. A. Lu, J. Stankovic, C. Lu, R. Zhang, and Y. Lu, "Feedback performance control in software services," *Control Systems Magazine, IEEE*, vol. 23, no. 3, pp. 74–90, 2003.

VM	QoS at 0secs	SLO at 25secs	SLO at 115secs	SLO at 205secs
pc2	1000MHz	85tps	250tps	150tps
pc3	1000MHz	100tps	40tps	20tps

Table IV
VARIATIONS REQUESTED BY EXTERNAL AGENTS IN THE TARGET SERVICE LEVEL OBJECTIVE

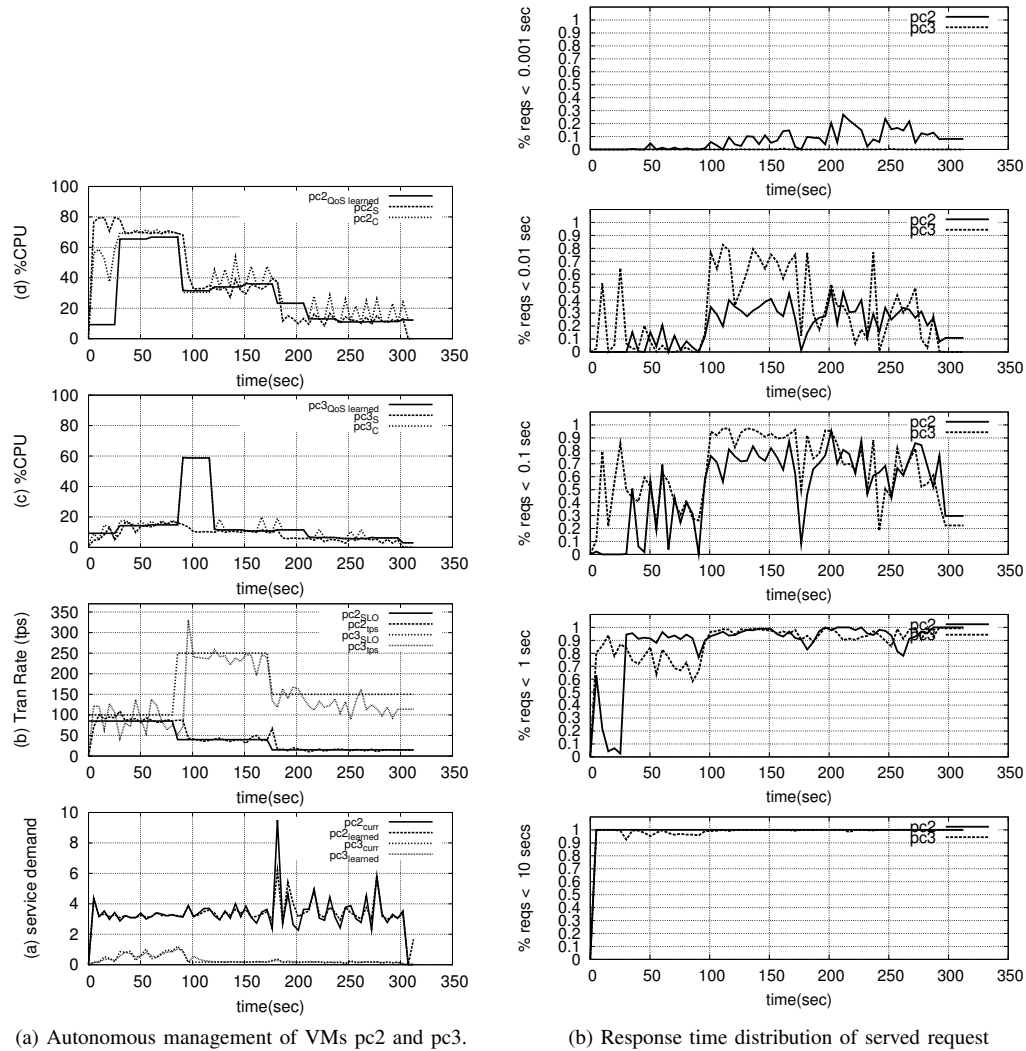


Figure 4. pc2 executes the web application server and pc3 executes the web image server. Http packets are controlled.

- [3] J. L. Hellerstein, "Challenges in control engineering of computing systems," IBM Research Division, Thomas J. Watson Research Center, P.O. Box 704, Research Report RC23159 (W0309-091), sep 2003.
- [4] D. A. Menasc, M. N. Bennani, and H. Ruan, "On the use of online analytic performance models in self-managing and self-organizing computer systems," in *Self-star Properties in Complex Information Systems*, ser. Lecture Notes in Computer Science, Ö. Babaoglu, M. Jelasity, A. Montresor, C. Fetzer, S. Leonardi, A. P. A. van Moorsel, and M. van Steen, Eds., vol. 3460. Springer, 2005, pp. 128–142.
- [Online]. Available: http://dx.doi.org/10.1007/11428589_9
- [5] Q. Zhang, L. Cherkasova, and E. Smirni, "A regression-based analytic model for dynamic resource provisioning of multi-tier applications," in *ICAC '07: Proceedings of the Fourth International Conference on Autonomic Computing*. IEEE Computer Society, 2007, p. 27.
- [6] L. Bertini, J. C. B. Leite, and D. Mosse, "Statistical qos guarantee and energy-efficiency in web server clusters," in *ECRTS '07: Proceedings of the 19th Euromicro Conference*

on Real-Time Systems. Washington, DC, USA: IEEE Computer Society, 2007, pp. 83–92.

- [7] B. Urgaonkar and A. Chandra, “Dynamic provisioning of multi-tier internet applications,” in *ICAC '05: Proceedings of the Second International Conference on Automatic Computing*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 217–228.
- [8] C.-H. Tsai, K. G. Shin, J. Reumann, and S. Singhal, “Online web cluster capacity estimation and its application to energy conservation,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 7, pp. 932–945, 2007.
- [9] I. M. Llorente, R. S. Montero, E. Huedo, and K. Leal, “A grid infrastructure for utility computing,” in *WETICE '06: Proceedings of the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 163–168.
- [10] D. A. Menascé and E. Casalicchio, “Qos in grid computing,” *IEEE Internet Computing*, vol. 8, no. 4, pp. 85–87, 2004.
- [11] D. Colling, T. Ferrari, Y. Hassoun, C. Huang, C. Kotsokalis, S. McGough, Y. Patel, E. Ronchieri, and P. Tsanakas, “On Quality of Service Support for Grid Computing,” in *2nd International Workshop on Distributed Cooperative Laboratories and Instrumenting the GRID (INGRID 2007)*, April 2007. [Online]. Available: <http://pubs.doc.ic.ac.uk/ingrid-2007/>
- [12] F. Rodríguez, F. Freitag, and L. Navarro, “On the Use of Intelligent Local Resource Management for Improved Virtualized Resource Provision,” in *The 2nd Workshop on System-level Virtualization for High Performance Computing (Eurosys 2008)*. Glasgow, UK: ACM DL, March 2008.
- [13] Siege, “An http regression testing and benchmarking utility,” 2008, <http://www.joedog.org/JoeDog/Siege>.
- [14] Image_3D, “A pear package for rendering 3d objects,” 2008, http://pear.php.net/package/Image_3D/.